2000

Impacts of Distributed Simulation on Electronic Warfare Test Results

Major Darrell L. Wright
JADS Joint Test Force
2050A 2nd St. S.E.
Kirtland Air Force Base, New Mexico 87117-5522
505-846-1015
wright@jads.kirtland.af.mil

Captain Sandra K. Smith
JADS Joint Test Force
2050A 2nd St. S.E.
Kirtland Air Force Base, New Mexico 87117-5522
505-846-0462
smiths@jads.kirtland.af.mil

Clyde J Harris
Science Applications International Corporation
JADS Joint Test Force
2050A 2nd St. S.E.
Kirtland Air Force Base, New Mexico 87117-5522
505-846-0909
harris@jads.kirtland.af.mil

Keywords:
advanced distributed simulation, electronic warfare, test and evaluation, high level architecture, latency

ABSTRACT: *The Joint Advanced Distributed Simulation (JADS) Joint Test Force was chartered by Office of the Secretary of Defense to investigate the utility of advanced distributed simulation (ADS) technology to test and evaluation (T&E). JADS executed three test programs (command, control, communications, computers, intelligence, surveillance and reconnaissance; precision guided munitions; and electronic warfare [EW]) representing slices of the overall T&E spectrum as well as observing other activity within the T&E community to form its conclusions. One of the slices, the Electronic Warfare Test, used the high level architecture in the creation of a synthetic environment to recreate a series of open air range flight test events. This paper discusses the JADS EW Test results with particular emphasis on the performance of the synthetic environment, the impacts of ADS on the EW Test results, and the lessons learned.*

1. Introduction

The Joint Advanced Distributed Simulation (JADS) Electronic Warfare (EW) Test was the first high-performance test and evaluation (T&E) federation built using the high level architecture (HLA). This paper describes the two versions of the federation, how we instrumented the federation, how we integrated the federation, presents the execution results, and ends with a discussion of the test result implications for T&E and HLA.

2. JADS Electronic Warfare Test Description

The JADS Joint Test Force (JTF) completed the final phase of its HLA-based EW Test in April 1999. This was the last phase in a series of tests designed to investigate the utility of advanced distributed simulation (ADS) technology for test and evaluation and to address T&E community concerns with the HLA. Test participants were the Air Force Electronic Warfare Evaluation Simulator (AFEWES), Fort Worth, Texas; the U.S. Navy Air Combat Environment Test and Evaluation Facility (ACETEF), Patuxent River Naval Air Station, Maryland; and the JADS JTF Test Control and Analysis Center (TCAC), Albuquerque, New Mexico.

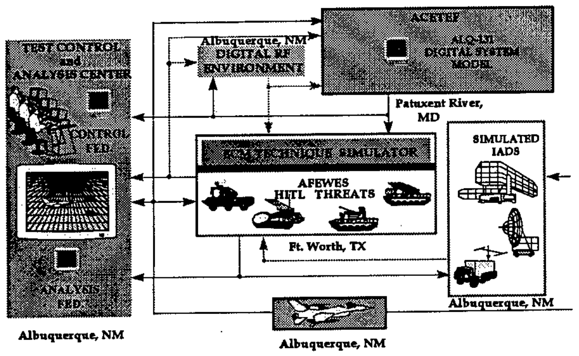
DISTRIBUTION STATEMENT A
Approved for Public Release
Distribution Unlimited

20000608 059

DTIC QUALITY INSPECTED 4

This was the second ADS-based test for the JADS EW Test team. The first occurred in December 1998. It used the same distributed architecture and test participants with a personal computer (PC)-based digital system model (DSM) at ACETEF in place of the actual self-protection jammer (SPJ). Both tests were designed to evaluate how ADS might enhance the test and evaluation process at different stages in EW system development and to provide the Department of Defense's EW community with another tool to overcome traditional test resource limitations.

In order to evaluate ADS for T&E users, we designed tests to recreate the EW performance data produced by traditional EW development tests for standard EW measures of performance. Our first test was a series of open air range (OAR) flight tests using the F-16 and the SPJ. Hardware-in-the-loop (HITL) testing using the AFEWES threat simulators and systems integration laboratory of the jammer supplemented the open air range testing data. These traditional EW tests provided JADS with the baseline data necessary for driving models simulating the F-16 flight profile over the range, background noise sources, and terminal threat cueing. Components of the JADS test are shown in figure 1. In addition, we used the baseline data to verify that the ADS architecture and the JADS federation were accurately recreating the environment seen in the traditional tests.



ECM = electronic countermeasures
IADS = Integrated Air Defense System
RF = radio frequency

Figure 1. JADS EW Test Components

2.1 Federation Design

The federates used for the JADS tests were of two types: playback federates and pass-through (gateway) federates. Playback federates used a general purpose HLA-compliant software tool designed to publish predefined scripts of data attributes and interactions to a federation at correct times during the simulation. JADS used the playback federates to replay important OAR component data recorded in Phase 1 - hence their designation. The playback data were loaded during the federation joining process and transmitted based on a timed sequence of events. Playback federates are platform, radio frequency (RF) environment, and terminal threat hand-off (TTH).

Pass-through federates are special purpose gateways that use the same HLA-compliant software as the playback federates. The difference is that instead of a script, the federate publishes data from a PC-based model or analysis tool. The pass-through federate is not a general purpose gateway since the PC-based software (model or analysis tool) communicated to the federate software through a unique communication structure. (Pass-through functionality was compliance tested separately.) In the EW federation, pass-through federates were responsible for publishing data generated by application software hosted on a PC and transmitting data subscribed from the JADS federation to the PC software application. Examples of pass-through federates are the jammer/DSM and test control facility (TCF).

The test scenario represented by each F-16 pass over the test range, or run, was reduced into time-ordered "data scripts" representing the time-space-position information (TSPI) of the aircraft, command and control threat hand-off commands, and the background RF environment. The interactions between the AFEWES threats and the DSM or jammer were dynamic based upon aircraft location, the AFEWES man-in-the-loop operations, and capabilities of the system under test (SUT) against a specific simulated threat. The JADS EW Test federation consisted of the federates described below.

- Test Control Facility (TCF) - The TCF managed the test execution and collection of necessary data to evaluate SPJ and ADS performance measures. The TCF interfaced with the automated data reduction software (ADRS) computers serving as pass-through federates to propagate the setup, start-up, and stop commands as well as performing other test control and display functions.

- Terminal Threat Hand-off (TTH) - The TTH federate was responsible for assigning terminal threats at the threats federate to the target simulated by the platform federate. This federate provided initial position data to the threats for target acquisition.

- Radio Frequency Environment (RFENV) - In order to replicate the test environment from the OAR test phase during the ADS phases, only two AFEWES threats were active for each run. The RFENV playback federate simulated the other two threat signals in the ADS test environment by supplying the mode activation/deactivation messages.

- Jammer/Digital System Model (DSM) - This pass-through federate was responsible for providing the simulation of the ALQ-131 jammer and its RF environment. During the Phase 2 ADS test, this simulation of the SUT consisted of an all software simulation of the jammer as well as the RF environment presented to it. In the Phase 3 ADS test, the actual jammer was inserted at ACETEF. The jammer was stimulated using threat waveforms generated by equipment at ACETEF. The waveform generators were controlled by switch actions made by the AFEWES threat operators. Commands were sent through the runtime infrastructure (RTI). ACETEF used a flexible gateway to connect the waveform generators and the jammer to the federation.

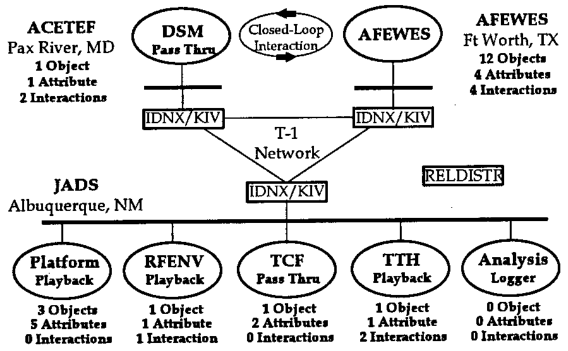
- Aircraft Platform - This playback federate provided the positional information for the test aircraft to the other federates. This aircraft position was based on Global Positioning System (GPS)-recorded data from the OAR test missions.

- Threats - This federate located at AFEWES provided the terminal threat simulations. These simulations include human-operated threat simulations designed to track a simulated target. Threats also provided an RF simulation of the SPJ technique waveforms. All these assets interfaced with the federation through an HLA gateway.

- Analysis - The analysis federate had many of the same display and analysis functions of ADRS; however, it took a different approach to data collection and scenario visualization. It is an example of a federate that acts as a logger for the federation.

2.2 Federation Configuration

The primary JADS tools for documenting detailed requirements were the Federation Execution Planners Workbook (FEPW) [1] and a JADS Federation Interface Control Document (ICD) [2]. The ICD provided further details to developers beyond the content of the FEPW including federate attributes and interactions, computers and communication equipment/software, and RTI services required. To summarize, the JADS EW Test federation used dedicated T-1 circuits, communications, and encryption devices to link JADS with its two key EW test facilities, AFEWES and ACETEF. Three network nodes interconnect the JADS federates. Four of the federates execute on dedicated Silicon Graphics, Inc. (SGI) O2 workstations in the JADS test control facility in Albuquerque, New Mexico. The DSM federate executes on an SGI O2 at ACETEF, and the threats federate executes on an SGI Challenge computer at the AFEWES facility. The network architecture and federate geographic locations are illustrated in figure 2. JADS established both secure voice and secure digital by using KIV-7 encryption devices throughout the wide area network (WAN). These devices permitted point-to-point transfer and verbal transmissions of classified information. Once JADS and AFEWES and JADS and ACETEF reached their respective security agreements, the WAN segments were operationally ready to handle classified data. The only difference between Phase 2 and Phase 3 ADS architectures was the representation of the SUT. The test scenario and all other federates were the same.



IDNX = Integrated Digital Network Exchange
RELDISTR = reliable distributor
T-1 = digital carrier used to transmit a formatted digital signal at 1.544 megabits per second

Figure 2. JADS EW Federation

3. Federation Instrumentation

The ADS design caused some interesting changes in the physical distance between the jammer and the threats. The key closed-loop interaction between the jammer and the threats on the test range occurred at distances less than 50 nautical miles. In the ADS architecture, those same components interact more than 1000 miles apart. For the ADS-based tests, JADS set a design limit of 500 milliseconds for transmission latency within that closed loop. Simply stated, we wanted no more than 250 milliseconds to elapse from the time that the threat operator activated a particular radar mode until the jammer (or DSM) received the energy (or message) representing that mode. Likewise the return path could have no more than 250 milliseconds. To measure this latency and all the factors that could be affecting it, JADS instrumented the federation. Below is a description of the instrumentation we used.

3.1 Global Positioning System Receiver Time Source

The GPS receiver located in each facility was the time source for all the federates within that facility. It provided time sent by the GPS satellite constellation. It provided time in standard Inter-Range Instrumentation Group (IRIG)-B, 1 megahertz (MHz), 5 MHz, and 10 MHz signal outputs for use by timing distribution systems.

3.2 Hardware Timing Cards

All the computers in the TCAC, at ACETEF, and the gateway computer at AFEWES contained hardware cards manufactured by either BanComm or True Time that were connected to the IRIG-B time code signal from a GPS receiver. Implementation of the time cards varied. Some systems used the card to set the system clock in the computer, other systems used time directly from the card, and one instrumentation system used a fixed interval timing pulse on the card along with a local oscillator and registers to maintain time. One notable limitation was observed on the ADRS computers. The PCs running ADRS contained BanComm cards. The ADRS software was a Windows 16-bit application but BanComm only had 32-bit drivers. Therefore, those PCs obtained their time from the PC's system clock, which was periodically set to GPS time by a BanComm utility program. In general, the preferred implementation is to read time directly from the time card. Even in the absence of an IRIG-B synch pulse, the time cards should have better local oscillator-driven clocks than typical computer system time clocks.

JADS used the hardware and software described above to ensure all federates were time synchronized. The initial goal was 1 millisecond; however, software tools were not available to measure accuracy at that level. In fact, testing time synchronization across the federation was more art than science. Even with time synchronization and the time cards implemented in all computers, we still found instances where time synchronization "slipped" affecting latency measurements. We removed 2 runs from the Phase 3 data set for this reason.

3.3 JADS RTI Interface Logger

The logger resided in the software interface between the federate and the RTI. It recorded all function calls to and from the RTI along with all of the function data parameters. For example, when the federate wanted to publish data, it called the RTI updateAttributeValues function. When the logger was linked with the federate, the federate called the logger updateAttributeValues function. The logger stored the function identification and parameter data in the log file buffer and then called the RTI updateAttributeValues function. When a log file buffer became full, it was written asynchronously to the log file and a new buffer was created. The logger was designed to minimize impact on the federate it was linked with. Simple versions of this type of logger can add as little as 20 lines of code to the federate. [3]

The combination of the interface logger and time synchronization across all computers is essential to measure latency and identify out-of-order data instances. Federation loggers (loggers that are a separate federate) are capable of recording data but can't be used to measure latency because of the out-of-order data issues discussed in paragraph 3.6.

3.4 Network Monitoring

A combination of in-house tools and commercially developed software products provided JADS with a near real-time limited capability to assess network performance and evaluate the integrity of data as they were being collected during EW testing. The various tools were used to help provide a clear picture of the network and to speed diagnostic and maintenance efforts during a test. Data on the network were not collected for post-test analysis during Phase 2. As is discussed below this was changed for Phase 3 based upon observations made during Phase 2 post-test analysis.

Spectrum™, a network analysis package developed by Cabletron Systems, utilized the simple network management protocol (SNMP) tools to monitor communications and network hardware. This allowed JADS personnel to see, in near real-time, the status of the long-haul links as well as the routers connecting remote sites. A thirty-second polling frequency was used to monitor the EW Test equipment. The SNMP tools also allowed JADS to monitor and record the bandwidth used on the T-1 links.

A companion "tool" for monitoring the routers was a simple dot matrix printer connected to the network. Each router had the ability to log failures on a printer. The printer was in the TCAC next to the test director. When the router detected problems it would print out an error message on the printer. The printer noise would signal the test director to check the message. Runs could either be continued or terminated at the test director's discretion.

Each federate sent "link health" check updates and displayed the information received from other federates. This was used during testing not only to determine the health of the each federate but as another view of the overall health of the network.

In addition, a simple utility used standard pings to display the status of various federation computers and presented the test team and networking personnel with the first indication that a problem existed with the network and/or the computers at each site. Called the "stoplight" tool, it presented a small green, yellow, or red on-screen graphic for each monitored link, based on the link's current status. If pings were delayed or dropped, status changed.

For some runs, data dropouts and high latency spikes were noticed in post-test analysis. Router statistics were examined in real-time (very intrusive) for a few runs to determine if the routers were the problem. There was no indication that the routers were dropping packets and latency could not be examined with the available data. It was determined that protocol analyzers (PC-based network sniffers) located on each local area network (LAN) segment at each node would have allowed JADS personnel to determine the causes of some of the data dropouts and high latencies. Sniffers were not used during Phase 2 because changes to the Ethernet architecture would be necessary at each site and the required numbers of sniffers were not available. Sniffers were incorporated into the Phase 3 architecture and installed between the router and Ethernet switch.

3.5 Federate Monitoring

For the Phase 2 test, there were two types of periodic federate health checks. The tests showed that neither type provided completely satisfactory instrumentation for that purpose.

- RTI Heartbeat - The first health check was the internal RTI "heartbeat" message sent every six seconds via transmission control protocol (TCP)/internet protocol (IP) from each federate to the federation executive (FEDEX). If the FEDEX failed to detect three successive heartbeat messages from a federate, then it would display a warning message in the FEDEX window on the SGI O2 host RFENV federate.

- Federate Link Health Check - The federate "link health check" (LHC) system, as documented in the ICD, was the second health check. This system employed 1 hertz (Hz) LHC messages sent best effort from every federate to every other federate. It proved to be much more useful, both in real-time during the test runs and later during the post-test analysis, because of its higher frequency and the fact that the LHC messages were captured by the JADS RTI logger.

3.6 Instrumentation Within Data Structures

JADS elected to add instrumentation within the data structures to better understand latency and to guard against out-of-order data. In the user-defined header space of each message we added the time stamp that the federate creating the message deemed the message "valid." These data were then passed along with the message. Latency could quickly be determined from any subscriber's log file by comparing the log file time for message receipt to the time stamp in the message header. This also allowed data to be checked post-test for out-of-order delivery.

JADS was concerned about correcting out-of-order data delivery during execution. A sequence numbering scheme was specified in the ICD [2]. The plan was to allow federates to correct (if possible) out-of-order data during execution and to allow easy identification during data loss and out-of-order data during post-test analysis. During Phase 2, two of the federates implemented this incorrectly which caused problems with post-test analysis. Fortunately, we did not encounter problems with out-of-order data invalidating run results.

There are three types of out-of-order data that the federation designer must account for. For the remaining discussion keep the following picture in mind. A is a shooter, B is a shooter, and C is an observer.

The first case is unique to HLA. Because best effort and reliable data transmissions use different protocols, there is a difference in delivery latency. Assume A shoots and then transmits the projectile position. Furthermore, assume the projectile fired message is sent reliable and the position updates are sent best effort. Generally C will get one or more position updates before the projectile fired message. This is due to the difference in latency between the two protocols.

The second case is due entirely to the way networks behave. For this case assume that A, B, and C are connected via a WAN and are equally distant. No assumption is made about how the projectile fired messages are sent. In this case assume A fires at B, then B fires at A. Depending on how the network latencies align, C may see the scene correctly or it may see that B fires first. Latency is not just a product of distance. It is also a product of the relative processing and communications load at each site. This difference can cause messages from A to B to have less latency than messages from B to A. It can also cause messages from a close site to take longer than messages from a farther site.

The third case is more of a temporal perception flaw than that it truly causes out-of-order data. It is due to the way networks handle TCP/IP. The RTI does nothing to correct this behavior and, in some cases, can exaggerate the behavior. Assume A shoots at B. The shot is observed by C. Again, assume the projectile fired message is sent reliable. There is no guaranteed order of delivery for this message. Sometimes B will get the message first, sometimes C will get the message first. If B is not allowed to fire after A shoots, there is a real potential for B to fire after C gets the message from A, but before B received A's message. The RTI variant comes from how the reliable distributor (RELDISTR) works. The RELDISTR is an agent that passes reliable data to one or more subscribers. For this case, add another federate, D, to the scene. Assume that a single RELDISTR services both B and D. In this case it is easy to see that A's message could be sent to C and then to the RELDISTR that services B and D. The RELDISTR could then send the message to D before sending it to B thus increasing the time in which errors can occur. The problem can grow to seconds of difference when network collisions occur.

4. Federation Integration

Successful federations are simply a matter of systematic, common sense engineering. It begins with a detailed design documenting the federation interface. The document needs to clearly define all common data that will be passed prior to the federation execution as well as data that will be passed during the execution. Complex data like antenna patterns and platform signatures that are common to more than one federate are as critical to match as the coordinate transforms needed during execution. Likewise the scenario logic needs to be understood and documented. JADS elected to use an ICD [2] for this purpose because the Object Model Development Tool and the Federation Execution Planner's Workbook were not sufficiently mature to completely convey the information. The ICD became the bible we used during integration.

Questions that need to be answered in federation integration: "Did we build the federation right?" and "Did we build the right federation?" The first question looks at the components and the integration. Are all the federates and the federation ICD compliant? The second question looks at the suitability of the federation to meet the sponsors' needs. It should be no surprise that careful planning and documentation will not only answer the engineering questions, but also provide ammunition for the verification and validation of the federation. A detailed discussion of JADS integration approach and our lessons learned are presented in the following section.

4.1 JADS Integration Approach

JADS approach to integrating the hardware, software, and facility unique components (e.g., AFEWES HITL simulators, gateways) in preparation for both JADS ADS test phases was divided into several tasks. Each task was intended to incrementally assemble and test the architecture. The first task was to create a stable network and computer architecture. This included implementing the RTI and tuning the performance of the hardware and software components while the federates were being designed and coded. For Phase 2 it took eleven months of work using five RTI versions to reach a stable network architecture that was ready to support test execution. For Phase 3 it took one month to implement the change from RTI 1.3 release 4 to release 5.

Once the initial network architecture was configured and the benchmarks for the network and RTI were completed, JADS took initial delivery of the federates and began incremental integration, testing, and build up of the federation components. Two tests were critical to force timely completion of the effort. The first test series was acceptance testing in which each federate was tested for ICD compliance. This test uncovered minor discrepancies that JADS elected to live with. (One of these came back to cause significant data analysis problems. Two of the federates incorrectly implemented the sequence number scheme making it difficult to reconstruct some parts of the data transfers.) The second test really proved that the federation would provide the data JADS needed. That test was the federation integration test (FIT).

The FIT was conducted in five phases designed to provide the results required to support the formal JADS test readiness review. During the FIT a predetermined number of test runs were conducted daily based on scheduled test events. The number of test events coupled with the specific objectives gave the team the confidence that the actual test was executable and the federation would provide valid results. Prior completion of the acceptance tests meant that the software components were functional and adequate allowing the FIT to focus on the functionality and adequacy of the integrated software capability. The procedures for the FIT included static and dynamic tests organized under five test phases.

- Phase 1 - Network components and time synchronization verification
- Phase 2 - Federation components and functionality
- Phase 3 - Test control and monitoring capabilities
- Phase 4 - Federation execution with manned threat pairs for north- and south- bound runs (wet and dry)

- Phase 5 - Data collection, retrieval, and analysis capabilities

During the FIT, all test start-up, execution, and stop procedures were verified. Site coordination and status control were exercised. Federate and federation execution were fully demonstrated and verified. AFEWES federate software was fully exercised, and integration with other federate software was proven. The 84-run goal for the Phase 2 FIT was not achieved because of problems within AFEWES, but all threats were demonstrated. Post-test data processing and analysis capability were performed and verified.

During Phase 3 integration, JADS used the playback federate to inject data logged during the Phase 2 execution into the federation. This subjected the ACETEF gateway to realistic loading without the cost of using the full AFEWES facility. Phase 3 also used a highly streamlined version of the FIT since the execution and analysis of Phase 2 gave JADS the experiences needed to quickly sort and resolve the issues that arose.

4.2 Integration Lessons Learned

Several key lessons come from our experiences in federation integration.

- Integration testing and network tuning need to be accomplished at the expected data rates and sizes that the federation will likely see. Static tests using fixed rates and sizes do not fully test the architecture. Message sizes need to be mixed and federates should be checked for bursts of data.
- An ICD is critical to understanding not only data rates and sizes, but to ensure there is a common understanding of all data that will be passed or are common to two or more federates. The ICD can be more effective if simple test cases are developed to allow results to be easily checked. Common test tools for items like coordinate transforms can also help.
- Federate acceptance tests are an excellent forcing function to speed problem resolution. JADS used the tests as leverage to get the federate developers to solve ICD compliance issues and to establish a clean point to transition configuration management from the developer to JADS.
- Federate integration tests are essential to understanding that the federation is functioning correctly and will provide valid data. The federates can all be ICD compliant and the federation still produce invalid data.
- Planning and documenting acceptance and integration tests are a cost-effective way to verify and validate federate and federation performance.
- Because of Phase 2 test scheduling and integration workload, JADS chose to defer federate compliance testing and certification until after Phase 2 was completed. We found that the extensive federate acceptance and integration test process fully verified federate performance and readiness for formal JADS test events. Compliance testing would not provide any added value to JADS in preparing to execute formal EW Test events.

5. Phase 2 Federation Execution Results

Based on our lengthy network and federation integration, we were confident that the network would be fairly well behaved and produce reasonably consistent latency. We did not expect to lose many runs to excessive latency or data dropouts. We did encounter a problem where best effort data were lost or delayed. This was most noticeable in the 20 Hz aircraft position (TSPI) data stream. We worked with Defense Modeling and Simulation Organization's (DMSO) RTI developers to build a workaround using a fixed start-up sequence to resolve the problem. This problem turned out to be a bug that was fixed in RTI 1.3 release 5. Actual performance is discussed below.

Phase 2 produced 363 total runs over nine days. Of those runs, 95 were aborted. The primary cause was federate failures (83) followed by procedure problems (10). Network problems accounted for two lost runs. Eight runs were tagged for further analysis since they showed high closed-loop latency (> 500 milliseconds) which exceeded our design limit. These runs were allowed back into the valid test results when we determined that the measures of performance fit within the population as estimated by the "good" run measures of performance.

5.1 Latency

The RTI interface logger files were used to record the arrival and departure times of all published and subscribed complex data types for each federate. JADS analysts utilized software tools to summarize and compare log file contents and determine round-trip federation latency as well as node-to-node latency values for particular complex data types. A data packet "sniffer" installed to monitor the ACETEF federated traffic during six runs provided some additional insight into the latency issue.

The analysis of high latency was approached in two ways, one focused on node-to-node latency while the other focused on latency for just those federation messages deemed federation latency. Federation latency message types include three complex data message types: MS_Source_Mode_Change, SUT_Jammer_Tech_Com, and SUT_Receiver_Track_Update. Latency values were computed for these message types to travel "round trip" between the AFEWES and ACETEF federates.

For the first approach, node-to-node latency values across relevant network links were evaluated for six complex data message types. The six types were selected for evaluation based on their ability to provide insight into the impact of latent traffic on SUT data validity. In other words, these were the message types that if latent should have had the most noticeable effect on SUT behavior and the SUT performance measure data collected. Ten individual runs with unusually high node-to-node latency values, out of 246 completed runs, were marked and further studied for anomalies before being included in the SUT valid data set. Table 1 provides a summary of node-to-node latency, categorized by message type and network link.

For the second approach, analysts calculated "round-trip" federation latency values by summing the individual message latencies between AFEWES and ACETEF nodes for "latency critical" message types. Out of 246 successfully completed trial runs, eight experienced unsuitable round-trip federation latency values (> 500 milliseconds [ms]) and were marked for probable exclusion from the valid SUT data set. Calculated average and maximum round-trip federation latencies, based on the remaining successful runs, were 254 ms and 380 ms, respectively. Further analysis was performed on the marked runs to determine the potential causes and outcomes of extremely high (e.g., >13 seconds) latency values. Two causes were determined to be responsible for these extremes. The first, a network link or network equipment outage that caused reliable (TCP) data traffic to be held up on multiple occasions was deemed responsible for < 15% of the extreme latency values seen. The other, more serious problem was due to a design defect in the DSM federate software that has since been identified and fixed. During Phase 2 testing, an algorithm (i.e., Nagle algorithm) was in place in the code that caused some message traffic to wait and be "bundled" with later traffic before being distributed by the federate. Depending on the order and timing in which "latency critical" message types were bundled for distribution, they may have had to experience large wait times.

5.2 Bandwidth

Bandwidth was not a real factor in Phase 2 execution. The 56% utilization was due to a heavier ping rate and Spectrum sampling taken during a few runs to try and isolate latency and drop out behaviors we were seeing.

EW PHASE 2 - NETWORK LINK PERFORMANCE		
	BANDWIDTH UTILIZED	
NETWORK LINK	AVERAGE	PEAK
JADS - AFEWES	6.75%	27%
JADS - ACETEF	2.96%	56%
AFEWES - ACETEF	4.18%	19%

5.3 Data Loss

The RTI interface loggers collected published and subscribed complex data types for each federate. Real-time network instrumentation files and test observer notes were additional sources of information as to the potential cause of certain data losses.

The analysis of lost data was approached from both a number of lost messages standpoint and a time standpoint. For the first approach, lost messages were tallied for six complex data message types across relevant network links. The six types were selected for evaluation based on their ability to provide insight into the impact of lost traffic on SUT data validity. In other words, these were the message types that if lost should have had the most noticeable effect on SUT behavior and the SUT measure of performance data collected. For these message types, there were no TCP (reliable) data traffic losses, and <1% of all user datagram protocol (UDP) (best effort) data traffic were lost. Thirty-eight individual runs with unusual data losses (> 5 messages lost) were marked and further studied for anomalies before being included in the SUT valid data set. Table 2 provides a summary of lost data traffic categorized by message type and network link.

For the second approach, analysts detected all data losses longer than 1 second and attempted to categorize the cause and outcome of each data loss using a combination of observer notes and test instrumentation. The outcome of individual data losses was typically one of two extremes; either the loss had essentially no observable impact on SUT performance measure data, or the run had to be aborted. This data loss outcome was dependent on several factors including the duration of the outage and the associated number, type, destination, and importance of the lost message packets. The timing of the outage during the run was noted to have an impact as well. For instance, an 8- to 10-second TSPI data loss during a run could be overcome by federation software dead reckoning algorithms, while a similar loss at the beginning of a run, before the transfer of any TSPI data, could not be handled and resulted in the run being aborted. The cause of each data loss event turned out to be a tougher analysis problem. Instrumentation sources included network equipment self-diagnostic error message files, observed real-time "ping" traffic outages, and real-time network load and packet rate query results; yet, even with this

Table 1. Node-to-Node Traffic Latency by Data Element (milliseconds)

DATA ELEMENT	TYPE	JADS-AFEWES	JADS-ACETEF	AFEWES-ACETEF
Live Entity State (Aircraft TSPI)	UDP	Avg: 43.9 Max: 859	Avg: 41.2 Max: 861	N/A
Threat Performance (Threat Track Data)	UDP	Avg: 45.9 Max: 860	Avg: 42.6 Max: 861	N/A
Threat Performance (T/E, Jamming-to-Signal Ratio, Target Location)	UDP	Avg: 32.1 Max: 7975	N/A	Avg: 35.7 Max: 8540
SUT_Jammer _Tech (DSM RF Emissions)	TCP	N/A	Avg: 130 Max: 13951	Avg: 104.3 Max: 9680
SUT_Receiver_Track (Verify Environment)	TCP	N/A	Avg: 112.7 Max: 13982	N/A
Source_Mode Change (Threat RF Emission)	TCP	Avg: 101 Max: 9556	Avg: 66 Max: 8022	Avg: 61 Max: 7701

Table 2. Lost Data Traffic Messages by Link

DATA ELEMENT	TYPE	JADS-AFEWES	JADS-ACETEF	AFEWES-ACETEF
Live Entity State (Aircraft TSPI)	UDP	Avg Lost: 6.9 Avg Sent: 4000 Max: 503	Avg Lost: 18.6 Avg Sent: 4000 Max: 1266	N/A
Threat Performance (Threat Track Data)	UDP	Avg Lost: 6.3 Avg Sent: 4000 Max: 504	Avg Lost: 18.1 Avg Sent: 4000 Max: 1266	N/A

Threat Performance (T/E Jamming-to-Signal Ratio, Target Location)	UDP	Avg Lost: 4.2 Avg Sent: 4000 Max: 182	N/A	Avg Lost: 14.8 Avg Sent: 4000 Max: 2222
SUT_Jammer _Tech (DSM RF Emissions)	TCP	N/A	Avg Lost: 0 Avg Sent: 9 Max: 0	Avg Lost: 0 Avg Sent: 9 Max: 0
SUT_Receiver_Track (Verify Environment)	TCP	N/A	Avg Lost: 0 Avg Sent: 96 Max: 0	N/A
Source_Mode Change (Threat RF Emission)	TCP	Avg Lost: 0 Avg Sent: 50 -90 Max: 0	N/A	Avg Lost: 0 Avg Sent: 50 - 90 Max: 0

instrumentation many of the data losses that occurred during Phase 2 remained impossible to attribute definitively to network links, network equipment, the RTI, or federation activity. In most cases, especially those where the data loss duration was short, there just was not conclusive evidence to determine the outage cause.

5.4 Unexpected Behavior

Aside from the unexpectedly high latency and data loss discussed above, JADS found several disturbing problems as we began to probe deeper into the data. Analysis showed several facets about the latency and the data loss events. In the JADS design, we used a single RELDISTR to service the federates in Albuquerque. This is not the default. Examination of the latency for reliable messages sent to two federates in Albuquerque showed that there could be a difference as large as 4 seconds. (Remember that these federates are all in the same location and are receiving the message from the same RELDISTR. This shows that the third type of out-of-order data discussed in paragraph 3.6 can be quite large.) Distance was not a consistent factor in the latency observed. One message from AFEWES took 10 seconds longer to reach Albuquerque than it did to reach Patuxent River. We also found that data dropouts could be seen where other message types using the same transport mechanism were still being passed.

This last finding is more interesting considering what we learned about the RTI after Phase 2 execution. First, we found that effectively all the JADS best effort data were passed in a single multicast group. We discovered this analyzing the few runs where we added a network sniffer at ACETEF. Discussions with DMSO uncovered that the runtime infrastructure initialization data (RID) file sets the maximum number of multicast groups. Our RID file set the maximum number at four with the last channel being a broadcast channel. We also learned that the channels are set by the join sequence.

The second thing we learned about the RTI ties back to the four-second latency seen in reliable messages within JADS. What appears to be a single RELDISTR is actually three RELDISTR. Departing from the default requires the user to change the RID file to tie each federate to a specific RELDISTR. We didn't do that, and, as a result, some of the problems we observed may have been due to this.

6. Phase 3 Federation Execution Results

Phase 3 brought four significant changes to the architecture. First, we modified the routers by adding more memory and using more efficient software. Second, as mentioned earlier, we changed from RTI 1.3 release 4 to 1.3 release 5. We found this product to be far more stable than release 4. Third, we changed the RID file to lock down the RELDISTR that serviced the federates located in Albuquerque at JADS. These changes coupled with the improved integration tools discussed in paragraph 4.1 indicated to us that we had a much more stable architecture. Finally, we added network sniffers at each geographic location to better isolate problems that we might find.

The sequence number problem from Phase 2 was fixed in the two offending federates. This coupled with the sniffer data collection tools gave us full insight into the messages that were being passed. Federation build-up and integration testing indicated that the latency and data loss problems seen in Phase 2 were gone. Expected latency was down and data loss was down to a few isolated messages per run.

Phase 3 produced 270 total runs over nine days. Of those runs, 32 were aborted. The primary cause was again federate failures (21). This time, however, most of the failures were in non-HLA components within the facilities. Again procedure problems (10) were the second most common cause of aborted runs. Network problems accounted for one lost run. Additionally, one run was tagged for further analysis because of high closed-loop latency (> 500 milliseconds). Overall performance of the federation was more stable and left very few mysteries to solve.

6.1 Latency

The study of Phase 3 latency was approached using the same tools and methods as for Phase 2; the only difference being the employment of data packet "sniffers" at all three sites during all Phase 3 runs. Analysis of "sniffer" data post-test shed insight into the behavior of the RTI software, federate software, and network in transporting data packets and helped analysts pinpoint the cause of anomalous latencies.

As in Phase 2, the analysis of high latency was approached in two ways: one focused on node-to-node latency while the other focused on just those federation messages deemed federation latency.

For the first approach, node-to-node latency values across relevant network links were evaluated for the same six complex data message types used in Phase 2. Eight individual runs with unusually high node-to-node latency values, out of 223 completed runs, were marked and further studied before being included in the SUT valid data set. Table 3 provides a summary of node-to-node latency categorized by message type and network link.

For the second approach, analysts calculated "round-trip" federation latency values by summing the individual message latencies between AFEWES and ACETEF nodes for "latency critical" message types. Out of 223 successfully completed runs, only one run experienced unsuitable round-trip federation latency values (> 500 ms) and was marked for probable exclusion from the valid SUT data set. Calculated average and maximum round-trip federation latencies based on the remaining successful runs were 164 ms and 417 ms, respectively.

Further analysis was performed on the marked runs to determine the potential causes of the high latency values experienced. With the help of the "sniffer" data, several causes were identified. The first, also experienced during Phase 2, was a simple network link or network equipment outage. The impact of a short duration network link drop was a loss of best effort

Table 3. Node-to-Node Traffic Latency by Data Element (milliseconds)

DATA ELEMENT	TYPE	JADS-AFEWES	JADS-ACETEF	AFEWES-ACETEF
Live Entity State (Aircraft TSPI)	UDP	Avg: 45.7 Max: 1150	Avg: 44.4 Max: 472	N/A
Threat Performance (Threat Track Data)	UDP	Avg: 52.7 Max: 1151	Avg: 52.3 Max: 516	N/A
Threat Performance (T/E, Jamming-to-Signal Ratio, Target Location)	UDP	Avg: 30.0 Max: 515	N/A	Avg: 41.2 Max: 511
SUT_Jammer _Tech (DSM RF Emissions)	TCP	N/A	Avg: 75.3 Max: 312	Avg: 67.3 Max: 296
SUT_Receiver_Track (Verify Environment)	TCP	N/A	Avg: 77.0 Max: 267	N/A
Source_Mode Change (Threat RF Emission)	TCP	Avg: 55.7 Max: 372	Avg: 71.9 Max: 1548	Avg: 45.5 Max: 501

traffic and a reliable traffic delay. Although this was the most infrequent cause of latent traffic, it was responsible for the most extreme Phase 3 latency values. A second cause of unusual latencies was the inconsistent treatment of the execution control stop messages by the various federates upon federation termination. On occasion, a federate, usually AFEWES, was able to continue sending data packets after the stop command was received, resulting in lost or latent messages to other federates. Stricter implementation of federation termination procedures in accordance with the ICD might alleviate this behavior. Third, incoming best effort messages were occasionally delayed by federation or RTI software after reaching the federate's LAN, inexplicably, perhaps due to some scheduling or system loading problem that preempted RTI processor control. These "within federate" message delivery problems were the second most frequent cause of latent data during Phase 3. However, the most frequent cause of latency problems in Phase 3 was the unusual behavior of the ACETEF reliable distributor in delivering outgoing messages to the other federates. Messages sent by the ACETEF reliable distributor experienced differential latencies to their destinations because of sequential outgoing packets being differentially held up at the sending end. No explanation could be found for this unusual reliable distributor behavior.

6.2 Bandwidth

Bandwidth was not a factor in Phase 3 execution. Traffic patterns were consistent across test days, and link usage corresponded well to expected data traffic levels. The peak values contained occurred during excursion runs with data traffic passed for four active threats. Excluding these runs, peak values for each link are 19%, 10%, and 12% respectively.

EW PHASE 3 - NETWORK LINK PERFORMANCE		
	BANDWIDTH UTILIZED	
NETWORK LINK	AVERAGE	PEAK
JADS - AFEWES	5.53%	65%
JADS - ACETEF	2.73%	10%
AFEWES - ACETEF	2.51%	21%

6.3 Data Loss

The study of lost data for Phase 3 was approached using the same tools and methods as for Phase 2; the only difference was the employment of data packet "sniffers" at all three sites during all Phase 3 runs. Analysis of "sniffer" data post-test shed insight into the behavior of the RTI software, federate software, and network in transporting data packets and helped analysts find the cause of data losses as well as determine the location where the data were lost. Again, lost data were analyzed from both a perspective of number of messages lost and loss duration.

For the first approach, lost messages were tallied for six complex data message types across relevant network links. The six types were selected for evaluation based on their ability to provide insight into the impact of lost traffic on SUT data validity. In other words, these were the message types that if lost should have had the most noticeable effect on SUT behavior and the SUT performance measure data collected. For these message types, there was a single TCP (reliable) message loss event from ACETEF to both JADS and AFEWES; whereas no reliable traffic was lost during Phase 2. Phase 3 results for UDP (best effort) data traffic also differed greatly from Phase 2 in that far fewer of these messages were lost during Phase 3. Only two individual runs experienced any data losses of these message types during Phase 3 compared to thirty-eight runs with unusual data losses (> 5 messages lost) during Phase 2. Table 4 provides a summary of this lost data traffic categorized by message type and network link.

The two runs with UDP data losses were marked and further studied for anomalies before being included in the SUT valid data set, as was the run with the TCP data loss. The cause of the majority of the lost data was determined to be a short duration network equipment outage that impacted the JADS-AFEWES link for approximately one second. This same event contributed to some of the extreme data latency problems experienced. There was not conclusive evidence to identify the cause of the data loss event for the second run; only a single data packet was lost.

The circumstances surrounding the loss of the single reliable data message are far more interesting as TCP data traffic, by its very nature, should not have been lost. Analysis of the EtherPeek packet sniffer data for this event provided evidence to strongly suggest that the RTI itself was responsible for the lost ACETEF SUT_Jammer_Tech message to both AFEWES and JADS. The EtherPeek packet data collected at ACETEF confirmed that the message was transmitted; and the EtherPeek packet data collected at JADS and AFEWES showed that it was received by the RTI and acknowledged at both sites. However, it was not recorded by the loggers at either destination, indicating that for some reason the RTI simply did not pass the message content on to the federates. This run was an excursion run during which four threat systems were active instead of two; yet, network tools did not indicate any problems or errors or even excessive traffic levels at the time the message was sent. The data from this run were not included in the valid SUT data set.

For the second approach, analysts detected all data losses longer than 1 second and attempted to categorize the cause and outcome of each data loss using a combination of observer notes and test instrumentation. Since there were so few data loss events during Phase 3, this technique revealed only one other strange data loss occurrence. On multiple occasions, best effort Link_Health data messages, transmitted by ACETEF before the start of the run, were sent via different IP multicast groups, one of which did not forward the messages to their destination. Typically, the ACETEF federate was the last to join the federation on occasions where these losses occurred. There was no possibility of any SUT data impact from these Link_Health message losses, but further study may indicate that anomalies in the federation joining process could cause best effort message traffic losses.

Table 4. Lost Data Traffic Messages by Link

DATA ELEMENT	TYPE	JADS-AFEWES	JADS-ACETEF	AFEWES-ACETEF
Live Entity State (Aircraft TSPI)	UDP	Avg Lost: .08 Avg Sent: 3900 Max Lost: 17	Avg Lost: 0 Avg Sent: 3900 Max Lost: 0	N/A
Threat Performance (Threat Track Data)	UDP	Avg Lost: .08 Avg Sent: 3800 Max Lost: 17	Avg Lost: 0 Avg Sent: 3800 Max Lost: 0	N/A
Threat Performance (T/E, Jamming-to-Signal Ratio, Target Location)	UDP	Avg Lost: .2 Avg Sent: 3900 Max Lost: 36	N/A	Avg Lost: .002 Avg Sent: 3900 Max Lost: 1
SUT_Jammer _Tech (DSM RF Emissions)	TCP	N/A	Avg Lost: .006 Avg Sent: 10 Max Lost: 1	Avg Lost: .006 Avg Sent: 10 Max Lost: 1
SUT_Receiver_Track (Verify Environment)	TCP	N/A	Avg Lost: 0 Avg Sent: 116 Max Lost: 0	N/A
Source_Mode Change (Threat RF Emission)	TCP	Avg Lost: 0 Avg Sent: 29 Max Lost: 0	N/A	Avg Lost: 0 Avg Sent: 29 Max Lost: 0

Router and RTI upgrades made prior to Phase 3 execution are given credit for the significant reduction in the amount of data lost in the phase over the previous one.

6.4 Unexpected Behavior

This time there were fewer unexpected behaviors. We again noted significant differences in latency for reliable messages sent to two federates in Albuquerque. (Again, recall that these are being serviced by the same RELDISTR.) However, the maximum was reduced to a few hundred milliseconds. We also found two runs where time synchronization was lost during the run. We found this by plotting latency as a function of message sequence number for selected message types.

Adding the sniffers to the instrumentation allowed us to look into what messages were being passed across the network. We found two interesting phenomena using this tool. First, we learned how complex the join process really was. Each federate join generates many messages over several seconds. On a few runs we allowed the analysis federate to join late and found that had a significant impact on latency. So much so that we recommend federations wait several seconds before beginning to execute and that high performance federations not allow federates to join in the middle of execution.

The second phenomenon deals with the transmission of reliable data. Our sniffer data revealed that all reliable data were being transmitted to all RELDISTRs even if the federate(s) that the RELDISTR serviced did not subscribe to these data. In subsequent discussions with DMSO we found that the transmission is based on "stream identification" which can be controlled using data distribution management.

7. Conclusions

HLA is evolving so rapidly that many lessons learned become obsolete with the next RTI or tool release. Below are our most relevant and enduring findings.

- ADS-based testing using HLA and the RTI is possible. The Federation Development and Execution Process (FEDEP) provides a good roadmap for ADS-based test development. We recommend using the FEDEP to guide and tailor your internal management and systems engineering processes.

- Instrumentation for ADS-based testing is not limited to the instrumentation placed on the system under test. Equal attention must be paid to the architecture itself. In-depth message monitoring (e.g., network sniffers) is essential to resolve problems. Real-time test control requires link health monitoring. Look for tools within the RTI or build them into the federation data structures and displays.

- Latency measurements require time synchronization, a logger between the federate simulation and the RTI, and a time stamp applied by the federate to indicate when the data were created/deemed valid.

- Systematic integration is necessary to prove the system will meet its intended purpose. Tools that help are an ICD, acceptance tests for each federate, and integration tests of the federation. It is cost effective to use these tests as part of the verification and validation (V&V) of the federation. HLA compliance testing will not reduce your integration requirement or time.

- Federate design and integration requires considerable cross-discipline knowledge. It is essential to have a team of experts that can address these areas:
  1. the system under test
  2. network and communications hardware and software
  3. platform operating system and communications
  4. tools and instrumentation
  5. federate software
  6. RTI software

This team must be lead by a strong systems engineer who will take responsibility for the federation from "lust to dust."

- Interface control and configuration management are crucial to success. Interface control begins with documenting the interface and all common data in a formal document. It includes the scenario logic - who, what, when, where, and why published events happen - as well as who cares or needs to know. Interface control continues with tests for conformance using common tools.

- Configuration management must control the following items: the ICD, the RTI version, the RID file, the federation file, the federation object model, the federate version, the federate platform hardware and operating system, the network hardware and software, and anything else that could affect the results if it were to change.

- RTI software can't be treated as a black box. Either buy expertise from the developer, establish a teaming arrangement, or make sure the documentation covers how the RTI implements the different communication protocols and how the performance of the RTI can be tuned to meet the needs of the federation.

JADS recommends that high-performance T&E federations (those with closed-loop transmission latency less than 500 milliseconds) use caution when implementing HLA. Extensive systems engineering and integration are essential. Performance using the current RTI family can be improved by using only best effort data transmission and by using multiprocessor computers exclusively. (We used Silicon Graphics/IRIX. This may not be necessary with other hardware/operating systems.)

Long-term investments are required to further improve the utility of HLA for the T&E community. These investments include multidimensional performance benchmark tools, version description documents for RTI releases to facilitate V&V and retesting when new RTI versions are installed in existing federations, RTI versions targeted toward true real-time operating systems and hardware, more efficient protocols to take advantage of forward error correction, message compression, and other advances.

8. References

[1] JADS FEPW
[2] The ICD version used for the JADS EW Test Phase 2 was Version 1.3, dated November 23, 1998.
[3] Jerrys paper about the logger

Author Biographies

MAJOR DARRELL L. WRIGHT is the Electronic Warfare Test team lead at the JADS Joint Test Force, Kirtland AFB, NM. He is a member of the HLA Architecture Management Group. He has a B.S. in physics and an M.S. in software system management. He has been on active duty in the Air Force since 1983.

CAPTAIN SANDRA K. SMITH is currently a member of the JADS Joint Test Force analysis team supporting both the End-to-End Test and Electronic Warfare Test. She received a B.A. from Miami University in Oxford, Ohio, and an M.S. in operations research from the Air Force Institute of Technology. She has been on active duty in the United States Air Force since 1989.

CLYDE J. HARRIS is a senior systems engineer for Science Applications International Corporation working with the JADS Joint Test Force. He has more than 27 years experience in business and military systems engineering.